Utility Processing

Read your *Adabas Utilities* documentation for specific information about changes to utilities for use in an Adabas cluster environment.

This chapter covers the following topics:

- ADADBS OPERCOM Commands
- ADADBS REFRESHSTATS - Refresh Statistical Values
- ADAICK PPTPRINT - Print/Dump Parallel Participant Table
- ADARAI - Adabas Recovery Aid
- ADAREP - Checkpoint Information Extended
- ADAPLP IPLOGPRI - Print Sequential Intermediate Data Sets
- ADARES CLCOPY - Copy/Merge Nucleus Cluster Command Logs
- ADARES PLCOPY - Copy/Merge Nucleus Cluster Protection Logs
- ADARES MERGE CLOG - Merge Nucleus Cluster Command Logs
- ADARES BACKOUT and REGENERATE— Uniquely Identifying Checkpoints
- ADASAV Processing Change
- ADASAV RESTPLOG -- Uniquely Identifying Checkpoints

ADADBS OPERCOM Commands

Changes have been made for ADADBS OPERCOM command processing in an Adabas Parallel Services cluster nucleus environment.

This section covers the following topics:

- Global Commands
- Routing a Command to a Specific Nucleus
- Routing a Command to All Cluster Nuclei

Global Commands

The following ADADBS OPERCOM commands have a GLOBAL option for making the request to all nuclei in an Adabas cluster:

- ADAEND

- CANCEL
- FEOFCL
- FEOFPL
- HALT

For example:

```
ADADBS OPERCOM ADAEND,GLOBAL
```

When GLOBAL is specified, the command is automatically propagated to all active cluster nuclei. When GLOBAL is *not* specified, a specific NUCID from the cluster must be specified and the command is sent to that NUCID.

Routing a Command to a Specific Nucleus

The NUCID option allows you to direct the OPERCOM commands to a particular nucleus in the cluster for execution.

The OPERCOM function's NUCID option is specified in a manner similar to the ADARUN function's NUCID parameter.

The following example sends the DSTAT command to the Adabas cluster nucleus designated with NUCID=3:

```
ADADBS OPERCOM DSTAT,NUCID=3
```

For inherently global commands, such as changing the setting of the TT parameter, the NUCID parameter is ignored.

Routing a Command to All Cluster Nuclei

When the NUCID option in the ADADBS OPERCOM function is not specified, the command is sent to all cluster nuclei and information is displayed for each nucleus in sequence.

ADADBS REFRESHSTATS - Refresh Statistical Values

The REFRESHSTATS function resets statistical values maintained by the Adabas nucleus for its current session. Parameters may be used to restrict the function to particular groups of statistical values.

In Adabas cluster environments, you must specify the specific nucleus (NUCID) for which statistical values are to be refreshed. If NUCID is not specified, statistical values will be refreshed for all active nuclei in the cluster.

ADAICK PPTPRINT - Print/Dump Parallel Participant Table

The PPTPRINT function has been added to the Adabas ADAICK utility to support an Adabas cluster environment. It is used to dump/print the parallel participant table (PPT) for the Adabas cluster.

Each of the 32 blocks (RABNs) allocated for the PPT represents a single nucleus in the cluster and comprises:

- a single header of fixed length; and
- multiple entries of variable length.

Note that in the dump/print, "PPH" is the tag for the PPT header and "PPE" is the tag for the PPT entries.

ADARAI - Adabas Recovery Aid

Adabas cluster products support the Adabas Recovery Aid (ADARAI).

ADARAI maintains a recovery log (RLOG) for each database; all nuclei in the cluster support a database write to the same RLOG and concurrent updates to the RLOG are controlled by a lock.

The ADARAI LIST function supports Adabas version 7 and above RLOGs; Adabas version 6 RLOGs are not supported.

ADAREP - Checkpoint Information Extended

Given that each cluster nucleus has its own PLOG data sets, checkpoints are no longer identified only by their name, PLOG number, and PLOG block number, but also by the ID of the nucleus that writes the checkpoint.

Several new parameters have been introduced for utilities that need to identify checkpoints on the PLOG.

ADAPLP IPLOGPRI - Print Sequential Intermediate Data Sets

The IPLOGPRI function is used to print the sequential intermediate data sets created from the PLOG merge process. Input to ADAPLP IPLOGPRI must be a MERGIN1/MERGIN2 data set created by the ADARES utility and specified in the JCL with DD name/link name DD/PLOG.

ADARES CLCOPY - Copy/Merge Nucleus Cluster Command Logs

Sample JCL has been added for allocating the intermediate data sets/files MERGIN1 and MERGIN2 required for automated CLOG copy/merge processing in nucleus cluster environments.

Notes:

1. When intermediate data sets/files are used for both CLCOPY and PLCOPY, the data set names must be unique so that they are not overwritten.
2. The data set BLKSIZE used must be greater than or equal to the largest CLOG BLKSIZE plus eight. The LRECL must be set to the BLKSIZE minus four.

- z/OS
- z/VSE
- BS2000/OSD

z/OS

```
//ALLOC JOB
//*
//* Example to allocate the ADARES CLCOPY intermediate data sets
//*
//CM1 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//OUTFIL DD DISP=(NEW,CATLG),SPACE=(CYL,(3,10)),UNIT=SYSDA,
// VOL=SER=volser,DCB=(RECFM=VB,LRECL=23472,BLKSIZE=23476),
// DSN=EXAMPLE.CLOG.MERGIN1
//INPFIL DD *
/*
//SYSIN DD *
REPRO INFILE(INPFIL) -
OUTFILE(OUTFIL)
/*
//*
//CM2 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//OUTFIL DD DISP=(NEW,CATLG),SPACE=(CYL,(3,10)),UNIT=SYSDA,
// VOL=SER=volser,DCB=(RECFM=VB,LRECL=23472,BLKSIZE=23476),
// DSN=EXAMPLE.CLOG.MERGIN2
//INPFIL DD *
/*
//SYSIN DD *
REPRO INFILE(INPFIL) -
OUTFILE(OUTFIL)
/*
```

z/VSE

```
//JOB ALLOC
//* Example to allocate the ADARES CLCOPY intermediate data sets
//EXEC PGM=IDCAMS
DEFINE CLUSTER (NAME (MERGIN1.CLCOPY) -
VOLUME(xxxxxx) CYLINDERS(3,10) RECSZ(23472) -
/*
//EXEC PGM=IDCAMS
DEFINE CLUSTER (NAME (MERGIN2.CLCOPY) -
VOLUME(xxxxxx) CYLINDERS(3,10) RECSZ(23472) -
/*
/&
```

BS2000/OSD

```
/BEGIN PROC A
/REMARK Example to allocate the ADARES CLCOPY intermediate files
/CREATE-FILE ADAddd.CLOG.MERGIN1,PUB(SPACE=(96,960))
/SET-FILE-LINK MERGIN1,ADAddd.CLOG.MERGIN1,BLKSIZE=STD(14)
/CREATE-FILE ADAddd.CLOG.MERGIN2,PUB(SPACE=(96,960))
```

```
/SET-FILE-LINK MERGIN2,ADAddd.CLOG.MERGIN2,BLKSIZE=STD(14)
.
.
/END-PROC
```

ADARES PLCOPY - Copy/Merge Nucleus Cluster Protection Logs

In an Adabas nucleus cluster environment, the protection logs (and optionally, the command logs: see the ADARES MERGE CLOG function and the ADARUN CLOGMRG parameter) of all individual nuclei in the cluster must be merged into single log files in chronological order for the cluster database shared by all the nuclei as a whole. The chronological order is determined by timestamps on all individual nucleus log records.

Protection logs are automatically merged when an ADARES PLCOPY is executed. In an Adabas cluster environment, the PLCOPY process accesses the parallel participant table (PPT) to determine which protection logs to copy and opens the appropriate data sets/files using dynamic allocation. PLCOPY copies/merges as much data as possible; if a nucleus is still writing to a protection log data set, PLCOPY 'partially' merges the data set.

The merge begins with the lowest timestamp from all protection logs being merged and ends with the lowest of the ending timestamps from all data sets/files. Records beyond this point are written to an 'intermediate' data set, which must be supplied as input to the subsequent merge. A cross-check ensures that the correct intermediate data set has been supplied.

ADARES expects that at least one of the protection logs being merged is at 'completed' status. If this is not the case, ADARES reports that there is no data to be copied.

A sample user exit (USEREX2P for dual logs or UEX12 for multiple logs) is provided to illustrate the necessary change for the intermediate data set.

- z/OS Sample Jobs
- z/VSE Sample Jobs
- BS2000 Sample Jobs

z/OS Sample Jobs

This section describes the sample jobs provided for z/OS systems.

ALLOC Job

The following sample JCL illustrates the allocation of the intermediate data sets MERGIN1 and MERGIN2 which are required for automated PLOG copy/merge processing in nucleus cluster environments.

Notes:

1. When intermediate data sets are used for both CLCOPY and PLCOPY, the data set names must be unique so that they are not overwritten.
2. The data set BLKSIZE used must be greater than or equal to the largest PLOG BLKSIZE plus eight.

The LRECL must be set to the BLKSIZE minus four.

```
//ALLOC JOB
//*
//* Example to allocate the ADARES PLCOPY intermediate data sets
//*
//CM1 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//OUTFIL DD DISP=(NEW,CATLG),SPACE=(CYL,(3,10)),UNIT=SYSDA,
// VOL=SER=volser,DCB=(RECFM=VB,LRECL=23472,BLKSIZE=23476),
// DSN=EXAMPLE.PLOG.MERGIN1
//INPFIL DD *
/*
//SYSIN DD *
REPRO INFILE(INPFIL) -
OUTFILE(OUTFIL)
/*
//*
//CM2 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//OUTFIL DD DISP=(NEW,CATLG),SPACE=(CYL,(3,10)),UNIT=SYSDA,
// VOL=SER=volser,DCB=(RECFM=VB,LRECL=23472,BLKSIZE=23476),
// DSN=EXAMPLE.PLOG.MERGIN2
//INPFIL DD *
/*
//SYSIN DD *
REPRO INFILE(INPFIL) -
OUTFILE(OUTFIL)
/*
```

ADARESPM Job

A sample job ADARESPM is provided on the JOBS data set to illustrate the manual execution of the PLCOPY merge function. Two intermediate data sets must be supplied. ADARES analyzes the data sets to determine which is to be used as input and which for output. Specific cross-checks determine whether the correct intermediate data set has been supplied; if not, ADARES will not continue. Continuing without the correct input can result in lost updates and inconsistencies if the output is used for REGENERATE or BACKOUT functions.

Once DDPLOGRn statements have been supplied on the session startup JCL, you do not need to supply them again for ADARES as these are opened using dynamic allocation. If the DD statements are supplied, they are ignored.

The following sample JCL illustrates the ADARES PLCOPY merge function:

```
//ADARESPM JOB
//*
//* ADARES : COPY/MERGE DUAL/MULTIPLE PROTECTION LOG
//* TWO COPIES OF OUTPUT ARE TO BE CREATED
//* FOR USE WITH AN ADABAS NUCLEUS CLUSTER
//*
//RES EXEC PGM=ADARUN
//STEPLIB DD DISP=SHR,DSN=ADABAS.Vvrs.LOAD
//*
//DDASSOR1 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.ASSOR1
//DDDATAR1 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.DATAR1
//MERGIN1  DD DISP=SHR,DSN=EXAMPLE.PLOG.MERGIN1
//MERGIN2  DD DISP=SHR,DSN=EXAMPLE.PLOG.MERGIN2
//DDSIAUS1 DD DSN=EXAMPLE.DByyyyy.PLOG1(+1),
```

```
// VOL=SER=ADAxxx,UNIT=TAPE,DISP=(NEW,CATLG)
//DDSIAUS2 DD DSN=EXAMPLE.DByyyyy.PLOG2(+1),
// VOL=SER=ADAxxx,UNIT=TAPE,DISP=(NEW,CATLG)
//DDDRUCK DD SYSOUT=X
//DDPRINT DD SYSOUT=X
//SYSUDUMP DD SYSOUT=X
//DDCARD DD *
ADARUN PROG=ADARES,MODE=MULTI,SVC=svc,DEVICE=3380,DBID=yyyyy
/*
//DDKARTE DD *
ADARES PLCOPY TWOCOPIES
/*
```

ADARES PLCOPY NOPPT—Ignore PPT

NOPPT is for emergency use when the PPT has been overwritten or is otherwise unavailable. It specifies that the PLOG data sets of all cluster nuclei are being supplied with DD names DDPLOGnn in the JCL.

Warning:
Use this parameter cautiously since it ignores the PPT and all control-type information typically coming from the PPT.

When you use this parameter, you must supply

- the correct intermediate data set; and
- the correct input protection logs from all nuclei with DD names DDPLOG01-nn.

The optional parameter SBLKNUM can be used to specify the starting block number for the sequential merge output.

Warning:
Without the PPT, ADARES cannot perform any extensive validations on the input data sets

ADARESIP Job

The following sample JCL illustrates the ADARES PLCOPY NOPPT merge function:

```
//ADARESIP JOB
//*
//* ADARES : COPY/MERGE DUAL/MULTIPLE PROTECTION LOGS FROM ALL
//* NUCLEI IN AN ADABAS NUCLEUS CLUSTER
//* PPT IS TO BE IGNORED
//* THIS IS ONLY FOR EMERGENCY USE WHEN THE PPT HAS BEEN
//* OVER-WRITTEN - USE CAUTION WHEN SUBMITTING
//*
//RES EXEC PGM=ADARUN
//STEPLIB DD DISP=SHR,DSN=ADABAS.Vvrs.LOAD <=== ADABAS LOAD
//*
//DDASSOR1 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.ASSOR1 <=== ASSO
//DDDATAR1 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.DATAR1 <=== DATA
//DDPLOG01 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.PLOGR1.NUC1 <=== PLOG1 NUC1
//DDPLOG02 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.PLOGR2.NUC1 <=== PLOG2 NUC1
//DDPLOG03 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.PLOGR1.NUC2 <=== PLOG1 NUC2
//DDPLOG04 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.PLOGR2.NUC2 <=== PLOG2 NUC2
```

```
//DDPLOG05 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.PLOGR1.NUC3 <=== PLOG1 NUC3
//DDPLOG06 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.PLOGR2.NUC3 <=== PLOG2 NUC3
//MERGIN1  DD DISP=SHR,DSN=EXAMPLE.PLOG.MERGIN1 <=== INTERMEDIATE 1
//MERGIN2  DD DISP=SHR,DSN=EXAMPLE.PLOG.MERGIN2 <=== INTERMEDIATE 2
//DDSIAUS1 DD DSN=EXAMPLE.DByyyyy.PLOG1(+1), <=== PLOG COPY
// VOL=SER=ADAxxx,UNIT=TAPE,DISP=(NEW,CATLG)
//DDDRUCK DD SYSOUT=X
//DDPRINT DD SYSOUT=X
//SYSUDUMP DD SYSOUT=X
//DDCARD DD *
ADARUN PROG=ADARES,MODE=MULTI,SVC=svc,DEVICE=3380,DBID=yyyyy
/*
//DDKARTE DD *
ADARES PLCOPY NOPPT
/*
//
```

z/VSE Sample Jobs

This section describes the sample jobs provided for z/VSE systems.

ALLOC Job

The following sample JCL is provided for allocating the intermediate data sets MERGIN1 and MERGIN2 which are required for automated PLOG copy/merge processing in nucleus cluster environments.

Notes:

1. When intermediate data sets are used for both CLCOPY and PLCOPY, the data set names must be unique so that they are not overwritten.
2. The data set block size used must be greater than or equal to the largest PLOG block size plus eight. The record length must be set to the block size minus four.

```
// JOB MRGFILE
// ASSGN SYS004,SYSIPT
// ASSGN SYS005,DISK,VOL=VSE209,SHR
// DLBL UOUT,'QA.DB1010.MERGIN1',0
// EXTENT SYS005,VSE209,1,0,10575,300
// EXEC PGM=OBJMAINT
./ CARD DLM=$$
./ COPY
$$
/*
// ASSGN SYS004,SYSIPT
// ASSGN SYS005,DISK,VOL=VSE209,SHR
// DLBL UOUT,'QA.DB1010.MERGIN2',0
// EXTENT SYS005,VSE209,1,0,11625,300
// EXEC PGM=OBJMAINT
./ CARD DLM=$$
./ COPY
$$
/*
/&
```

ADARESPM Job

A sample job ADARESPM is provided to illustrate the manual execution of the PLCOPY merge function. Two intermediate data sets must be supplied. ADARES analyzes the data sets to determine which is to be used as input and which for output. Specific cross-checks determine whether the correct intermediate data set has been supplied; if not, ADARES will not continue. Continuing without the correct input can result in lost updates and inconsistencies if the output is used for REGENERATE or BACKOUT functions.

Once PLOGRn statements have been supplied on the session startup JCL, you do not need to supply them again for ADARES as these are opened using dynamic allocation. If the PLOGRn statements are supplied, they are ignored.

The following sample JCL illustrates the ADARES PLCOPY merge function:

```
// JOB ADARESPM
//*
//* ADARES : COPY/MERGE DUAL/MULTIPLE PROTECTION LOG
//* TWO COPIES OF OUTPUT ARE TO BE CREATED
//* FOR USE WITH AN ADABAS NUCLEUS CLUSTER
//*
// LIBDEF PHASE,SEARCH=(SAGLIB.ASM742,SAGLIB.ADA742)
// DLBL ASSOR1,'EXAMPLE.DByyyyy.ASSOR1'
// DLBL DATAR1,'EXAMPLE.DByyyyy.DATAR1'
// DLBL MERGIN1,'MERGIN1.PLCOPY'
// DLBL MERGIN2,'MERGIN2.PLCOPY'
// TLBL SIAUS1,'EXAMPLE.DByyyyy.PLOG1'
// TLBL SIAUS2,'EXAMPLE.DByyyyy.PLOG2'
// ASSGN SYS021,TAPE
// EXEC ADARUN,SIZE=ADARUN
PROG=ADARES,MODE=MULTI,SVC=svc,DEVICE=3380,DBID=yyy
yy
/*
ADARES PLCOPY TWOCOPIES
/*
/&
```

ADARES PLCOPY NOPPT—Ignore PPT

NOPPT is for emergency use when the PPT has been overwritten or is otherwise unavailable. It specifies that the PPT is to be ignored and that the PLOG data sets of all cluster nuclei are being supplied with link names PLOGnn in the JCL.

Warning:
Use this parameter cautiously since it ignores the PPT and all control-type information typically coming from the PPT.

When you use this parameter, you must supply

- the correct intermediate data set; and
- the correct input protection logs from all nuclei with link names PLOG01-nn.

Warning:
Without the PPT, ADARES cannot perform any extensive validations on the input data sets.

ADARESIP Job

The following sample job ADARESIP is used with the ADARES PLCOPY NOPPT function:

```
// JOB ADARESIP
//*
//* ADARES : COPY/MERGE DUAL/MULTIPLE PROTECTION LOG
//* FOR USE WITH AN ADABAS NUCLEUS CLUSTER
//* CAUTION: NOPPT EXECUTION!
// LIBDEF PHASE,SEARCH=(SAGLIB.ASM742,SAGLIB.ADA742)
// DLBL ASSOR1,'EXAMPLE.DByyyyy.ASSOR1'
// DLBL DATAR1,'EXAMPLE.DByyyyy.DATAR1'
// DLBL PLOG01,'EXAMPLE.DByyyyy.PLOGR1.NUC1'
// DLBL PLOG02,'EXAMPLE.DByyyyy.PLOGR2.NUC1'
// DLBL PLOG03,'EXAMPLE.DByyyyy.PLOGR1.NUC2'
// DLBL PLOG04,'EXAMPLE.DByyyyy.PLOGR2.NUC2'
// DLBL PLOG05,'EXAMPLE.DByyyyy.PLOGR1.NUC3'
// DLBL PLOG06,'EXAMPLE.DByyyyy.PLOGR2.NUC3'
// DLBL MERGIN1,'MERGIN1.PLCOPY'
// DLBL MERGIN2,'MERGIN2.PLCOPY'
// TLBL SIAUS1,'EXAMPLE.DByyyyy.PLOG1'
// ASSGN SYS021,TAPE
// EXEC ADARUN,SIZE=ADARUN
PROG=ADARES,MODE=MULTI,SVC=svc,DEVICE=3380,DBID=yyyyy
/*
ADARES PLCOPY NOPPT
/*
/&
```

BS2000 Sample Jobs

This section describes the sample jobs provided for BS2000 systems.

ALLOC Job

For BS2000 systems, sample JCL is provided for allocating the intermediate files MERGIN1 and MERGIN2, which are required for automated PLOG copy/merge processing in nucleus cluster environments.

Notes:

1. When intermediate files are used for both CLCOPY and PLCOPY, the file names must be unique so that they are not overwritten.
2. The file BLKSIZE used must be greater than or equal to the largest PLOG BLKSIZE plus eight. The LRECL must be set to the BLKSIZE minus four.

```
/BEGIN PROC A
/REMARK Example to allocate the ADARES PLCOPY intermediate files
/CREATE-FILE ADAddd.PLOG.MERGIN1,PUB(SPACE=(96,960))
/SET-FILE-LINK MERGIN1,ADAddd.PLOG.MERGIN1,BLKSIZE=STD(14)
/CREATE-FILE ADAddd.PLOG.MERGIN2,PUB(SPACE=(96,960))
/SET-FILE-LINK MERGIN2,ADAddd.PLOG.MERGIN2,BLKSIZE=STD(14)
.
.
/END-PROC
```

ADARESPM Job

A sample job ADARESPM is provided on the JOBS file to illustrate the manual execution of the PLCOPY merge function. Two intermediate files must be supplied. ADARES analyzes the files to determine which is to be used as input and which for output. Specific cross-check determine whether the correct intermediate file has been supplied; if not, ADARES will not continue. Continuing without the correct input can result in lost updates and inconsistencies if the output is used for REGENERATE or BACKOUT functions.

Once DDPLOGRn statements have been supplied on the session startup JCL, you do not need to supply them again for ADARES as these are opened using dynamic allocation. If the file link statements are supplied, they are ignored.

The following sample JCL illustrates the ADARES PLCOPY merge function:

```
/BEGIN-PROC C
/MOD-TEST DUMP=YES
/REMARK *
/REMARK * A D A R E S COPY/MERGE DUAL PROTECTION LOG
/REMARK * 2 COPIES OF OUTPUT ARE TO BE CREATED
/REMARK * FOR USE IN A PARALLEL SERVICES
/REMARK * DATABASE
/REMARK *
/DEL-FI ADAddd.AUS1
/SET-JOB-STEP
/CREATE-FILE ADAddd.AUS1,SUP=TAPE(VOL=RES101,DEV-TYPE=T-C4),-
/ PROT=(USER-ACCESS=ALL-USERS)
/SET-JOB-STEP
/DEL-FI ADAddd.AUS2
/SET-JOB-STEP
/CREATE-FILE ADAddd.AUS2,SUP=TAPE(VOL=RES102,DEV-TYPE=T-C4),-
/ PROT=(USER-ACCESS=ALL-USERS)
/SET-JOB-STEP
/ASS-SYSLST L.RES.PAR
/ASS-SYSDTA *SYSCMD
/SET-FILE-LINK DDLIB,ADABAS.MOD
/SET-FILE-LINK BLSLIB00,ADAASM.MOD
/SET-FILE-LINK DDASSOR1,ADAddd.ASSO ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDDATAR1,ADAddd.DATA ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK MERGIN1,ADAddd.PLOG.MERGIN1 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK MERGIN2,ADAddd.PLOG.MERGIN2 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDSIAUS1,ADAddd.AUS1,ACC-METHOD=SAM,-
/ BUFF-LEN=32768,REC-FORM=V,SUP=TAPE(LABEL=STD)
/SET-FILE-LINK DDSIAUS2,ADAddd.AUS2,ACC-METHOD=SAM,-
/ BUFF-LEN=32768,REC-FORM=V,SUP=TAPE(LABEL=STD)
/START-PROG *M(E=ADARUN,L=ADABAS.MOD),RUN-MODE=ADV(A-L=YES)
ADARUN PROG=ADARES,DB=ddd
ADARES PLCOPY TWOCOPIES
/ASS-SYSDTA *PRIM
/ASS-SYSLST *PRIM
/END-PROC
```

ADARESIP Job

The following sample JCL illustrates the ADARES PLCOPY NOPPT merge function:

```
/BEGIN-PROC C
/MOD-TEST DUMP=YES
/REMARK *
/REMARK * A D A R E S COPY/MERGE DUAL PROTECTION LOGS
/REMARK * IN A PARALLEL SERVICES NUCLEUS
/REMARK * PPT IS TO BE IGNORED
/REMARK * THIS IS AN EMERGENCY USE FOR WHEN PPT
/REMARK * HAS BEEN OVERWRITTEN - USE CAUTION
/DEL-FI ADAddd.AUS
/SET-JOB-STEP
/CREATE-FILE ADAddd.AUS,SUP=TAPE(VOL=RES101,DEV-TYPE=T-C4),-
/ PROT=(USER-ACCESS=ALL-USERS)
/SET-JOB-STEP
/ASS-SYSLST L.RES.DELT
/ASS-SYSDTA *SYSCMD
/DEL-FI ADAddd.AUS
/SET-JOB-STEP
/CREATE-FILE ADAddd.AUS,SUP=TAPE(VOL=RES103,DEV-TYPE=T-C4),-
/ PROT=(USER-ACCESS=ALL-USERS)
/SET-JOB-STEP
/SET-FILE-LINK DDLIB,ADABAS.MOD
/SET-FILE-LINK BLSLIB00,ADAASM.MOD
/SET-FILE-LINK DDASSOR1,ADAddd.ASSO ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDDATAR1,ADAddd.DATA ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDWORKR1,ADAddd.WORK ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDPLOG01,ADAddd.PLOG1.NUC1 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDPLOG02,ADAddd.PLOG2.NUC1 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDPLOG03,ADAddd.PLOG1.NUC2 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDPLOG04,ADAddd.PLOG2.NUC2 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDPLOG05,ADAddd.PLOG1.NUC3 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDPLOG06,ADAddd.PLOG2.NUC3 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK MERGIN1,ADAddd.PLOG.MERGIN1,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK MERGIN2,ADAddd.PLOG.MERGIN2,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDSIAUS1,ADAddd.AUS,ACC-METHOD=SAM,-
/ BUFF-LEN=32768,REC-FORM=V,SUP=TAPE(LABEL=STD)
/START-PROG *M(E=ADARUN,L=ADABAS.MOD),RUN-MODE=ADV(A-L=YES)
ADARUN PROG=ADARES,DB=ddd
ADARES PLCOPY NOPPT
/ASS-SYSDTA *PRIM
/ASS-SYSLST *PRIM
/END-PROC
```

ADARES MERGE CLOG - Merge Nucleus Cluster Command Logs

In an Adabas cluster environment, command logs (CLOGs) from the cluster nuclei may be manually merged using the ADARES MERGE CLOG NUMLOG=nn function.

The NUMLOG parameter is required: it specifies the number of command log data sets/files to be included in the merge process. The maximum number is 32.

Sequential data sets/files are expected as input to the MERGE CLOG function; therefore, the ADARES CLCOPY function must be executed prior to the ADARES MERGE function.

The timestamp contained in the CLOGLAYOUT=5 format of the CLOG is required for the proper merging of command logs records.

- z/OS
- z/VSE
- BS2000

z/OS

The following sample job ADARESCM for z/OS systems (see the JOBS data set) illustrates the execution of the ADARES MERGE CLOG function:

```
//ADARESCM JOB
//*
//* ADARES : MERGE SEQUENTIAL COMMAND LOGS
//* FOR USE WITH AN ADABAS NUCLEUS CLUSTER
//*
//RES EXEC PGM=ADARUN
//STEPLIB DD DISP=SHR,DSN=ADABAS.Vvrs.LOAD <=== ADABAS LOAD
//*
//DDASSOR1 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.ASSOR1 <=== ASSO
//DDDATAR1 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.DATAR1 <=== DATA
//DDWORKR1 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.WORKR1 <=== WORK
//DDCLOG01 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.CLOGR1.NUC1 <=== CLOG1 NUC1
//DDCLOG02 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.CLOGR1.NUC2 <=== CLOG1 NUC2
//DDCLOG03 DD DISP=SHR,DSN=EXAMPLE.DByyyyy.CLOGR2.NUC3 <=== CLOG2 NUC3
//DDSIAUS1 DD DSN=EXAMPLE.DByyyyy.CLOGM, <=== OUTPUT OF
// VOL=SER=ADAxxx,UNIT=TAPE,DISP=(NEW,CATLG) CLOG MERGE
//DDDRUCK DD SYSOUT=X
//DDPRINT DD SYSOUT=X
//SYSUDUMP DD SYSOUT=X
//DDCARD DD *
ADARUN PROG=ADARES,MODE=MULTI,SVC=svc,DEVICE=3380,DBID=yyyyy
/*
//DDKARTE DD *
ADARES MERGE CLOG,NUMLOG=3
/*
//
```

z/VSE

The following sample job for z/VSE systems (see the JOBS data set) illustrates the execution of the ADARES MERGE CLOG function:

```
//JOB ADARESCM
//*
//* ADARES : MERGE SEQUENTIAL COMMAND LOGS
//* FOR USE WITH AN ADABAS NUCLEUS CLUSTER
//*
// LIBDEF PHASE,SEARCH=(SAGLIB.ASM742,SAGLIB.ADA742)
// DLBL ASSOR1,'EXAMPLE.DByyyyy.ASSOR1'
// DLBL DATAR1,'EXAMPLE.DByyyyy.DATAR1'
// DLBL WORKR1,'EXAMPLE.DByyyyy.WORKR1'
// DLBL ASSOR1,'EXAMPLE.DByyyyy.ASSOR1'
// DLBL CLOG01,'EXAMPLE.DByyyyy.CLOGR1.NUC1'
// DLBL CLOG02,'EXAMPLE.DByyyyy.CLOGR1.NUC2'
// DLBL CLOG03,'EXAMPLE.DByyyyy.CLOGR2.NUC3'
// TLBL SIAUS1,'EXAMPLE.DByyyyy.CLOGM'
// ASSGN SYS021,TAPE
// EXEC ADARUN,SIZE=ADARUN
ADARUN PROG=ADARES,MODE=MULTI,SVC=svc,DEVICE=3380,DBID=YYYYY
```

```
/*
ADARES MERGE CLOG,NUMLOG=3
/*
/&
```

BS2000

The following sample job for BS2000 systems (see the JOBS data set) illustrates the execution of the ADARES MERGE CLOG function:

```
/BEGIN-PROC C
/MOD-TEST DUMP=YES
/REMARK *
/REMARK * A D A R E S MERGE SEQUENTIAL COMMAND LOGS
/REMARK * FOR USE WITH PARALLEL DATABASE
/DEL-FI ADAddd.AUS
/SET-JOB-STEP
/CREATE-FILE ADAddd.AUS,SUP=TAPE(VOL=RES101,DEV-TYPE=T-C4),-
/ PROT=(USER-ACCESS=ALL-USERS)
/SET-JOB-STEP
/ASS-SYSLST L.RES.MERG
/ASS-SYSDTA *SYSCMD
/SET-FILE-LINK DDLIB,ADABAS.MOD
/SET-FILE-LINK DDASSOR1,ADAddd.ASSO ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDDATAR1,ADAddd.DATA ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDWORKR1,ADAddd.WORK ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDCLOG01,ADAddd.CLOG1.NUC1 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDCLOG02,ADAddd.CLOG1.NUC2 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDCLOG03,ADAddd.CLOG2.NUC3 ,SUP=DISK(SHARE-UPD=YES)
/SET-FILE-LINK DDSIAUS1,ADAddd.CLOGM,ACC-METHOD=SAM,-
/ BUFF-LEN=32768,REC-FORM=V,SUP=TAPE(LABEL=STD)
/START-PROG *M(E=ADARUN,L=ADABAS.MOD),RUN-MODE=ADV(A-L=YES)
ADARUN PROG=ADARES,DB=ddd
ADARES MERGE CLOG,NUMLOG=3
/ASS-SYSDTA *PRIM
/ASS-SYSLST *PRIM
/END-PROC
```

ADARES BACKOUT and REGENERATE— Uniquely Identifying Checkpoints

The protection log merge process usually changes the numbering of the PLOG blocks. The PLOG block number of a checkpoint on the original PLOG data set will not necessarily be the same as the block number after the merge. To uniquely identify the checkpoint in this situation, it is necessary to also specify the NUCID for all ADARES functions that can specify a TOBLK / FROMBLK parameter; that is, BACKOUT and REGENERATE.

The merge process ensures that there is at most one checkpoint per block. It records the (old) block number prior to the merge and the NUCID that wrote the checkpoint. When you then specify the block number and NUCID as reported in ADAREP, ADARES is able to uniquely identify the block.

Note:
In an Adabas nucleus cluster environment, ADAREP includes the NUCID when printing all checkpoint information.

The additional parameters that are required in an Adabas nucleus cluster environment are NUCID, TONUCID, and FROMNUCID.

If the NUCID is the same for the starting and ending checkpoint, only the NUCID needs to be specified.

Notes:

1. An ADAREP CPEXLIST function can be used to determine the original block number and NUCID that wrote the checkpoint. This is the block number prior to the merge and the one that ADARES REGENERATE and BACKOUT expects.
2. BACKOUT DPLOG and BACKOUT MPLOG are not allowed for a cluster database. The PLOG must be merged before the backout can be performed.

ADASAV Processing Change

Sample JCL is located in the ADASAVRW member of the JOBS data set.

For the following ADASAV functions, the Work data sets/files of all cluster (or noncluster) nuclei for the database that may have been active at the time of the ABEND must be reset:

- RESTONL (database)
- RESTONL GCB
- RESTORE (database)
- RESTORE GCB

This can be done either:

- manually (e.g., by using ADAFRM WORKRESET FROMRABN=1,SIZE=1B); or
- by specifying the Work data sets/files with DD names/link names DD/WORKRn (n=1-9) or DD/WORKnn (nn=10-32) in the JCL for the RESTONL/RESTORE function.

Otherwise, the nuclei that did not have their Work data sets/files reset will give parm-error 42 when started.

The DD/PLOGRn and DD/CLOGRn data sets are not reset in the restore process. They must be either copied/merged by ADARES PLCOPY/CLCOPY or reset by ADAFRM.

ADASAV RESTPLOG -- Uniquely Identifying Checkpoints

After the protection log merge process, the block number will not necessarily be the same. To uniquely identify the checkpoint in this situation, it is necessary to also specify the NUCID parameter for the ADASAV RESTPLOG function when specifying the SYN1 or SYN4 parameter.

Note:
An ADAREP CPEXLIST function can be used to determine the original block number and NUCID that wrote the checkpoint. This is the block number prior to the merge and the one that ADASAV RESTPLOG expects.